P-51

An Open Source Game Engine for Naval Education and Training

Presented to the

Navy Workforce Research and Analysis Conference

30 March 2004

Dr. Rudy Darken
Dr. Ted Lewis
Erik Johnson
Perry McDowell
Andrzej Kapolka
CDR Joe Sullivan

NAVAL POSTGRADUATE SCHOOL

Problem

Markets other than the gaming industry (e.g. Department of Defense) want access to gaming technologies

- i.e. console hardware, engines, content creation

... however ...

Gaming console market consists of closed stovepipes all with the same business model

- no opportunity for other markets (DoD) to benefit

Opportunity

Break the existing business model by supplying gaming hardware and software to non-traditional gaming customers

Commoditize gaming hardware and engines using open source and standards

Leverage first-mover advantage to dominate content creation and applications

- Order of magnitude speed-up in development time
- Order of magnitude decrease in development costs

Motivation

- The Navy wants the use of gaming technologies for education and training on an enterprise scale
- Invest once, standardize on an architecture, (a game engine) and enable reuse of content across many applications -- *independent of the application developer*

however

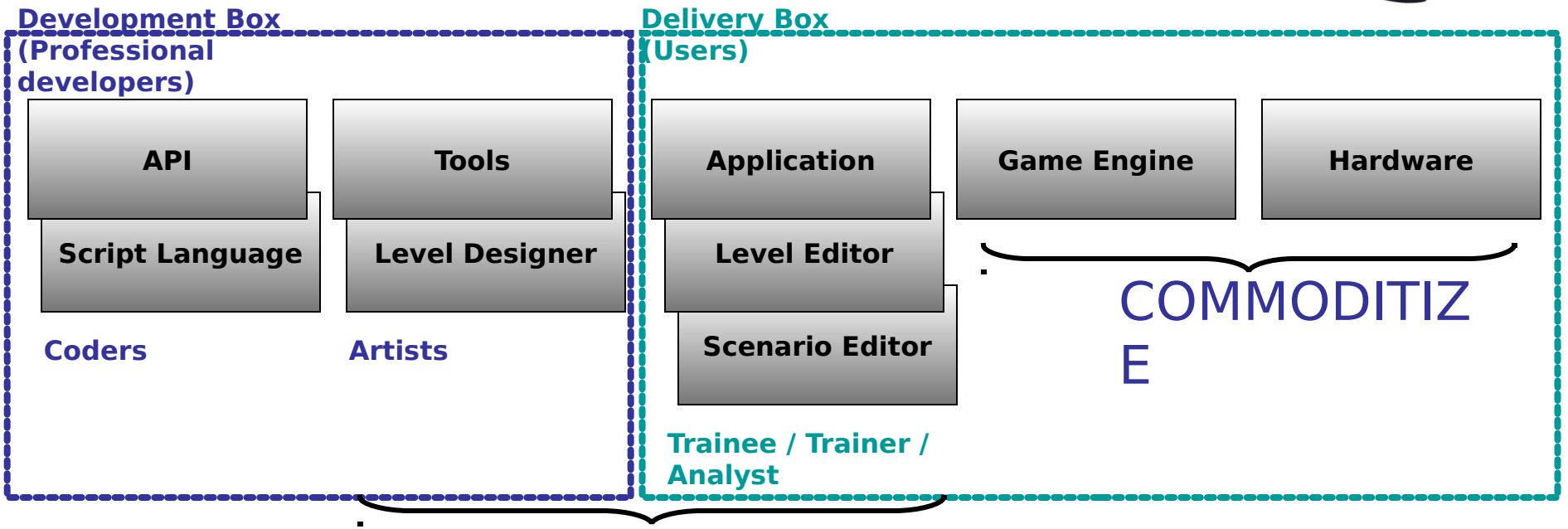
- The business model of the gaming market is based on profit from game titles -- this does not work for the Navy
- Game engines are extremely costly* and usage fees must be paid with each revision

* Can be anywhere from $300,000 to $1.5M for a full featured engine

P-51 Strategy

- Commoditize gaming engines and console hardware using open source and standards
- Leverage first-mover advantage to standardize content creation and applications
  - Order of magnitude speed-up in development time
  - Order of magnitude decrease in development costs
    - Pay non-recurring development costs ONCE
    - Pay recurring costs via maintenance model (e.g. Redhat)
- Motivate tool makers to participate
- Motivate prime contractors to adopt
- Motivate programs to endorse
- Motivate OSS community to contribute

Components



New business model is here

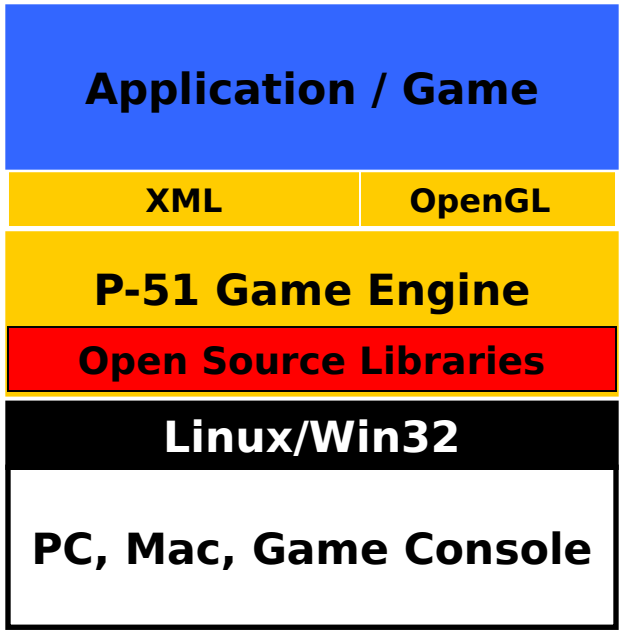
Only pay for content, not

The Product

P-51 Game Engine

- OpenGL (OpenSceneGraph)
- Linux / Win32 compatible
- Any Linux-compatible hardware
- XML Interfaces

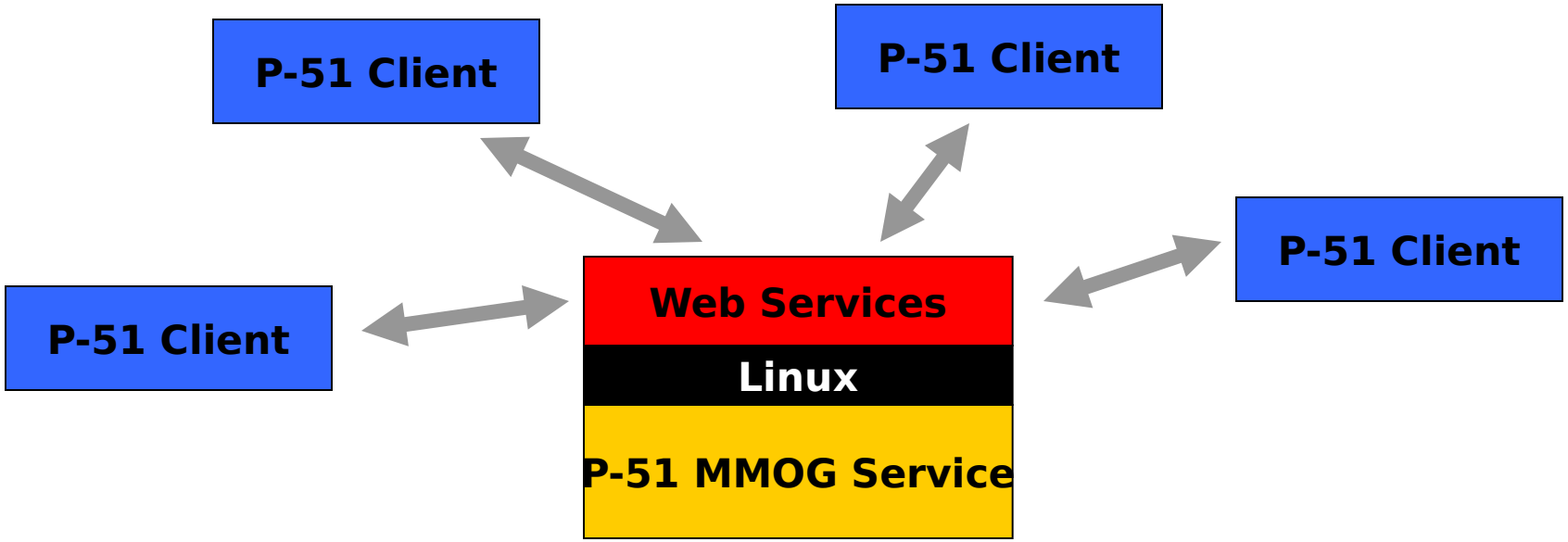
P-51 is a thin layer API that unifies several other open source libraries



Gentoo open source bootable Linux kernel

Networking

- Interoperability through web services
- Open architecture MMOG solution



Challenges

Maintenance Costs

- Funded annually, or ...
- Redhat model -- let developers pay for support as needed

Updates and Release Control

- Managed similar to how OSDL* manages Linux

Licensing Costs

- None -- P-51 is open and it stays open

Standardization

- Work with NIST to determine specification -- evolve over lifetime of P-51
- We will not create standards but will embrace and specify open standards for Navy use

Interoperability

- XRTI included (open source RTI: http://www.movesinstitute.org/~npsnet/xrti)
- MMOG features planned

Reusability

- All standard interfaces, XML, SOAP (for web services), OpenFlight
- All developers have access to all source code

* Open Source Development Labs (Linus Torvalds works there)

P-51 Status

Currently uses the following open source libraries:

- **FL**: GUI widgets
- **freetype**: Fonts
- **osg** (osgDB, osgGA, osgParticle, osgProducer, osgSim, osgText, osgUtil): Scene graph, file loading, particle systems
- **Producer**: Window handling, keyboard/mouse input, threads, timers
- **RTI-NG 1.3v6**: HLA communication
- **isense**: Tracker input
- **PLIB js**: Joystick input
- **PLIB sg**: Math types
- **PLIB ul**: Byte order conversion, other utilit
- **sigslot**: Boost signals-and-slots library
- **tinyxml**: XML parsing

Timeline and Costs

- Targeted release date for version 1.0
  - November 2004 (I/ITSEC)
- Prototype project
  - Shipboard fire fighting?
  - Intelligence simulations? (for CENNAVINTEL)
  - Homeland Security Games
- Estimated costs
  - Initial staffing
    - Four full-time P-51 engineers
      - $200,000 x 4 = $800,000
    - Four full-time artists and modelers
      - $175,000 x 4 = $700,000

Associated Risks

- Support from the fleet
  - P-51 will only succeed if it has buy-in from a large user base
- Self-supporting
  - P-51 must eventually be self-supporting. It should not rely on NETC or any other sponsor for recurring costs.
- Evolving standards
  - P-51 must be reviewed repeatedly for changes in requirements and standards. Is this compatible with P-51 being self-supporting?
  - We cannot create new standards (didn’t work for Ada and won’t work here either), but will specify P-51 to adopt open standards instead
- Separation of content from presentation
  - P-51 must support standard data file formats for separating content from the engine. Level editing must be included in the end product.

Business Case

Give away P-51 Game Engine (it stays open!)

Support, maintenance, training, supplied via Redhat model

Motivate non-Navy (and non-DoD) usage and participation -- share development costs through OSS

$1.5M to $2.5M over two years to steady state

- Run-time engine, fully developed API
- Linux kernel / Win32 compatible
- Development tools, level editors, content creation

Next steps

1. Specification -- NIST/NETC/NPS participation
2. Comparison to alternatives -- e.g. Epic Unreal
3. Prototype development
4. P-51 development and documentation

Why should I adopt P-51?

- No lock-in
  - Full open standard compliant
  - Maximize reusable content
- Cross platform
  - Runs on anything the Linux kernel runs on
  - Capable of running on game consoles
- No visible operating system
  - Ease of use
- Content creation tools

- Make it **flexible**, **hi-performance**, and **inexpensive**

Contact Information

Rudy Darken (Principal Investigator)
darken@nps.navy.mil
831-656-7588

Erik Johnson (Lead Engineer)
rejohnso@nps.navy.mil
831-656-2967

Perry McDowell (Content Developer)
mcdowell@nps.navy.mil
831-656-7591

Demo of Firefighting Trainer Built with P-51

Additional Information

Most of the information on the following slides was presented by other speakers at the Training Technology Session of NWRA. However, it is included here for completeness.

What does the M&S industry want?

- Low-cost, hi-performance hardware
- Ease of use … Marine-proof
- Scalability, interoperability
- Cheap run-time environments
  - No run-time licensing costs
- Powerful, inexpensive content creation
  - Rapid database development, scenario editing
- Stable tools and APIs
  - Reusable content

What does the Navy want?

- Compatibility with low-cost, hi-performance hardware
  - Off the shelf PCs and console components
- Ease of use ... Marine-proof
- Scalability, interoperability
- Cheap run-time environments
  - No run-time licensing costs -- massive deployment of applications
- Powerful, inexpensive content creation
  - Rapid database development, scenario editing
- Stable tools and APIs
  - Separation of content from presentation -- reusability!
- Low entry costs, and low recurring/maintenance costs
- Flexibility -- No vendor lock-in!
  - Standardize Navy-wide for maximum reuse

Ecosystem

How to beat Microsoft and Sony?

- Focus on niche market -- DoD M&S
  - Ground simulation?
- Motivate tool makers to participate
- Motivate prime contractors to adopt
- Motivate programs to endorse
- Motivate OSS community to contribute

Existing Business Models

TURNKEY SYSTEMS

FlightIG (MultiGen-Paradigm)
Evans & Sutherland
MetaVR
CG² Mantis

RUN-TIME TOOLS

MultiGen-Paradigm Vega
CG² VTree
SGI Performer
VisKit

GAME ENGINES / GAMES

Epic Unreal
Gamebryo (formerly NetImmerse) NDL

OPEN SOURCE

OpenSceneGraph
Crystal Space 3D
Nebula Device

CONTENT CREATION

IT Spatial
SimWright
Lockheed Martin, Boeing

CONTENT CREATION TOOLS

MultiGen-Paradigm Creator
Discreet 3D Studio Max
Terrex TerraTools

Existing Business Models

TURNKEY SYSTEMS

Pro: You get a full working system
Con: Any change will cost you. Full vendor lock-in

RUN-TIME TOOLS

Pro: Support, documentation
Con: Proprietary lock-in, run-time costs, licensing

GAME ENGINES / GAMES

Pro: Powerful, runs on cheap hardware
Con: High entry price, proprietary lock-in

Gam

OPEN SOURCE

Pro: Free!
Con: Poor support and documentation, usually low-level tools only

CONTENT CREATION

Pro: Can be open file formats
Con: High price, not applications

CONTENT CREATION TOOLS

Pro: Can be open file formats, absolutely necessary in the development process

Con:

These are part of the solution, but must be integrated to work efficiently